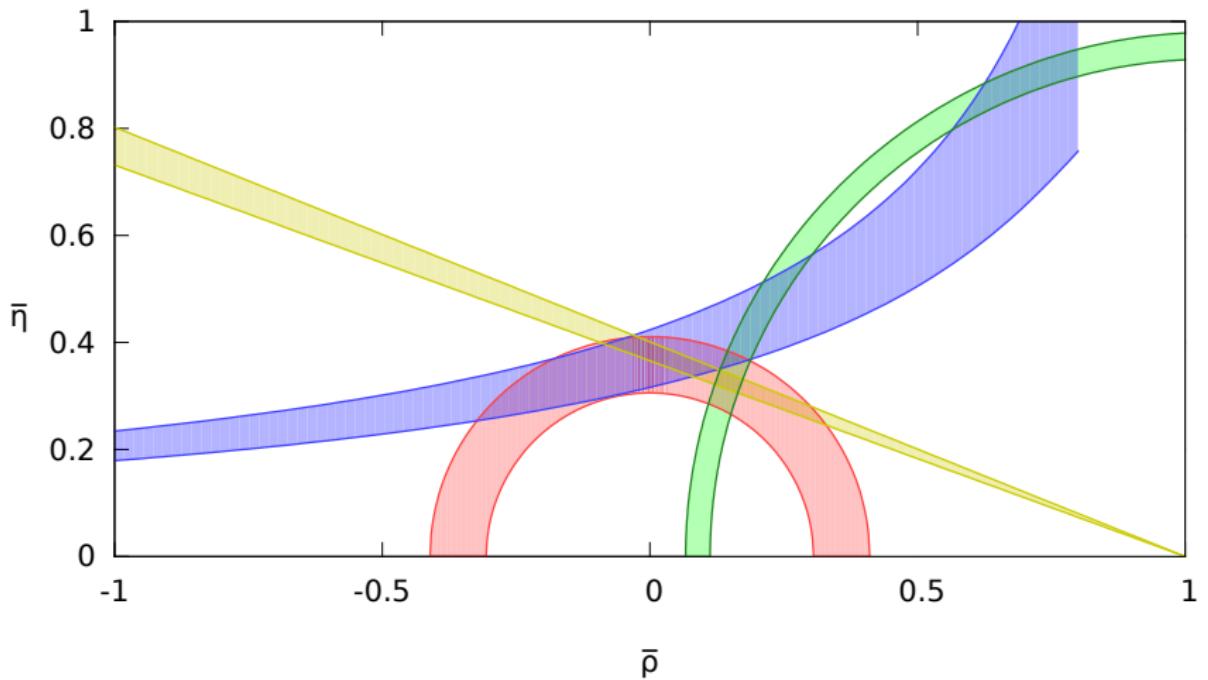
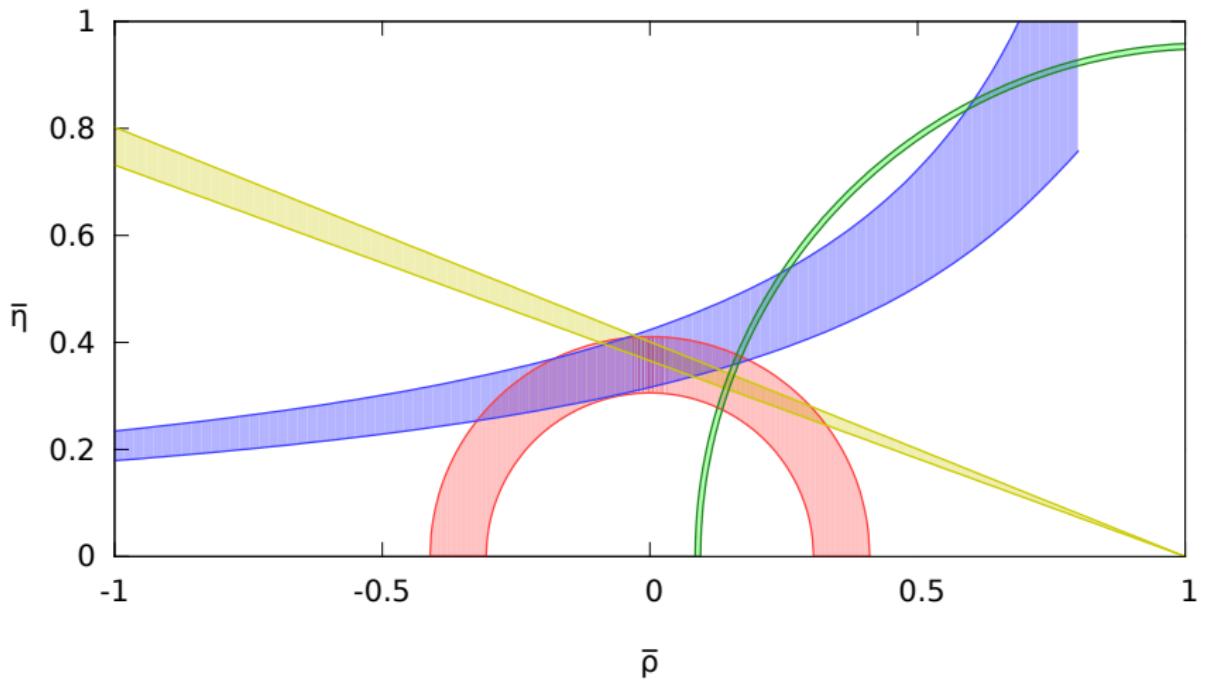


A new framework for perturbative calculations for a wide class of regulators

Christoph Lehner
BNL



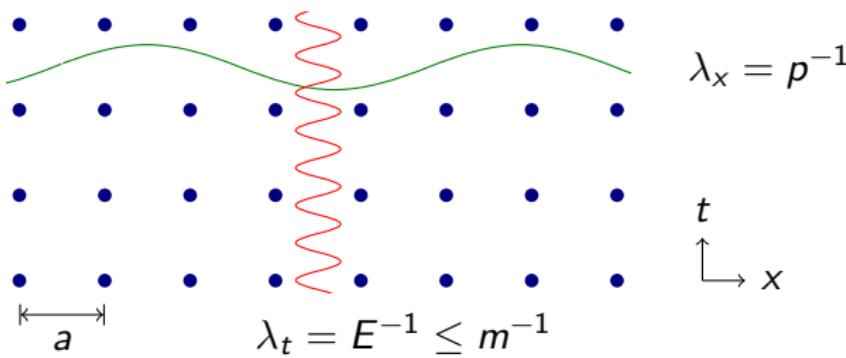
$\Delta M_s / \Delta M_d$
 $\sin 2\beta$
 $|V_{ub}/V_{cb}|$ (avg)
 $\epsilon_K + |V_{cb}|$



$\Delta M_s / \Delta M_d$ (No error in ξ)
 $\sin 2\beta$
 $|V_{ub}/V_{cb}|$ (avg)
 $\varepsilon_K + |V_{cb}|$

Simulation of heavy quarks on the lattice

- ▶ Problem: Heavy mesons “fall through the lattice”



- ▶ Mesons with mass m , momentum p , and energy $E=\sqrt{m^2+p^2}$
- ▶ Typical scales:
 $a^{-1} \approx 2 \text{ GeV}$, $m_D \approx 2 \text{ GeV}$, $m_B \approx 5 \text{ GeV} \Rightarrow am \geq 1$;
 $m_\pi \approx 0.2 \text{ GeV}$, $L = 32a \Rightarrow m_\pi L \approx 3$

Relativistic heavy quarks

(El-Khadra et al. 1997)
(S. Aoki et al. 2003) (Christ et al. 2006)

- ▶ Columbia formulation:

$$S = \sum_x \bar{Q}(x) \left((\gamma_0 D_0 - \frac{1}{2} D_0^2) + \zeta \sum_{i=1}^3 (\gamma_i D_i - \frac{1}{2} D_i^2) + m_0 + c_P \sum_{\mu, \nu=0}^3 \frac{i}{4} \sigma_{\mu\nu} F_{\mu\nu}(x) \right) Q(x)$$

- ▶ Tune coefficients of dimension 4 and 5 operators to remove $|a\vec{p}|$, $(am)^n$, $|a\vec{p}|(am)^n$ errors in on-shell quantities:

$$m_0, \zeta, c_P$$

- ▶ Matching of on-shell matrix elements of, e.g., HL operators requires $Q'(x) = Q(x) + d_1 \sum_{i=1,2,3} \gamma_i D_i Q(x)$ with parameter d_1 .

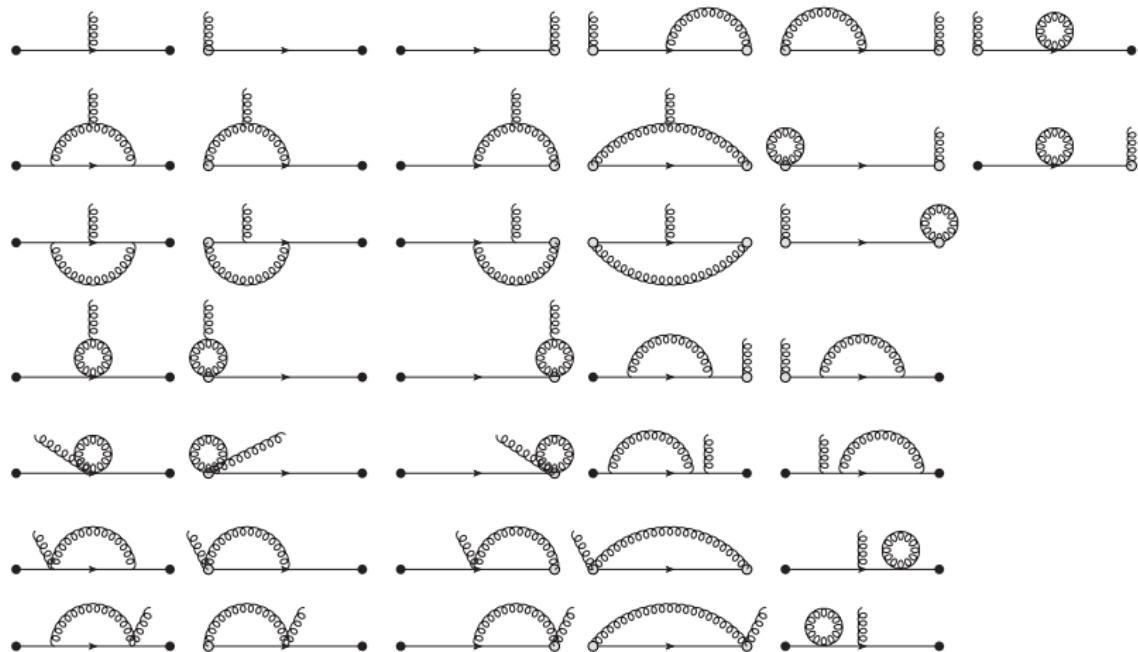
Calculations for a wide class of regulators

- ▶ The perturbative tuning of the lattice action,
- ▶ the suppression of lattice artifacts in lattice operators ($O(a)$ improvement),
- ▶ and the matching of lattice matrix elements to, e.g., $\overline{\text{MS}}$ -renormalized matrix elements

require the perturbative calculation of physical observables in different regulators.

Automate such calculations in a common framework

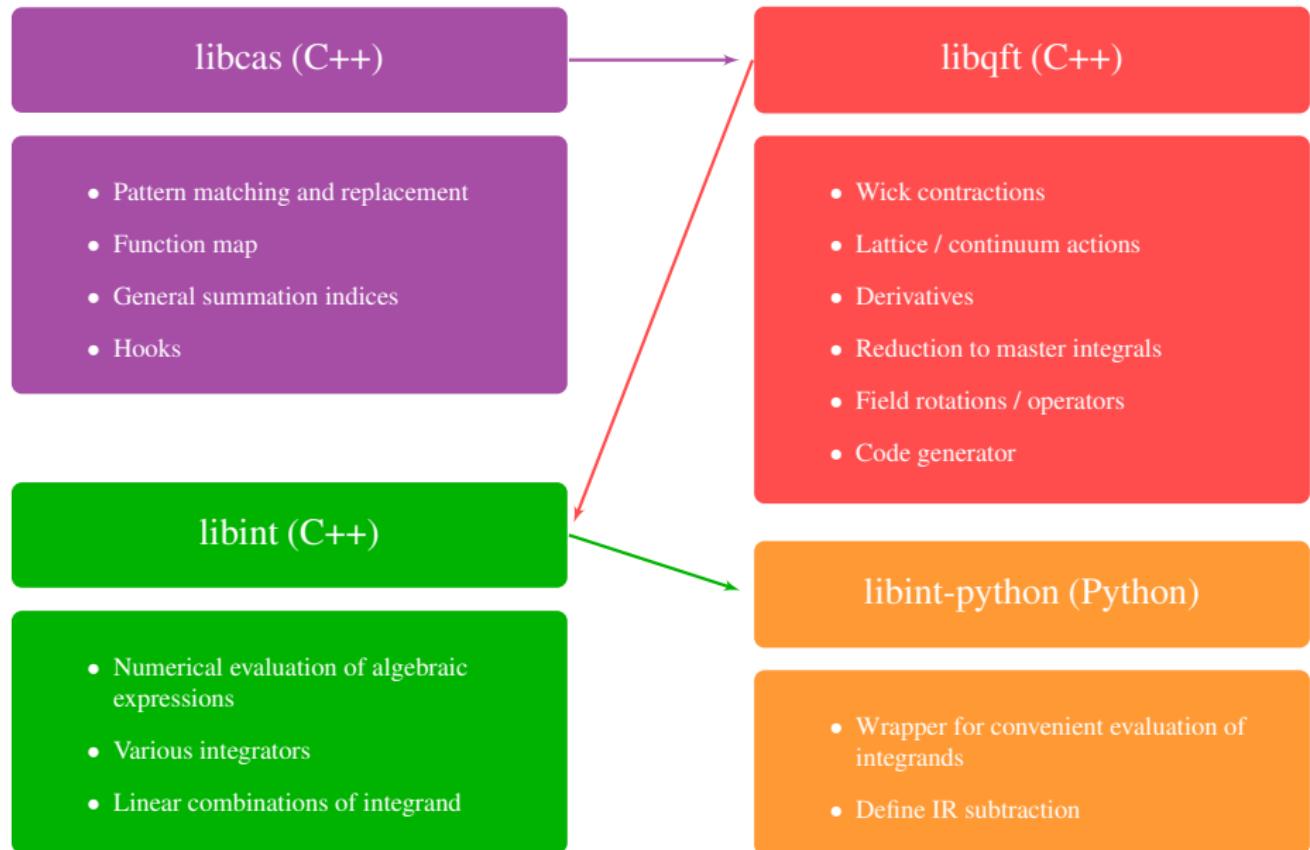
One-loop vertex graphs (RHQ)



There are more vertices in lattice regulators and they are more complex.

- ▶ Idea: Automate lattice PT analog to continuum PT in FORM-based approaches
- ▶ Problem: Simple expansion of terms prohibitive (complex vertices) \Rightarrow keep factorization (call graphs)
- ▶ FORM not suited to work on such call-graphs
- ▶ Wrote from scratch new computer algebra system (CAS) as a C++ library

On top of new CAS: unified LPT, continuum PT framework



Different classes of regulators

- ▶ 4-dimensional momentum space: Wilson, RHQ, Gauge, Continuum (d -dimensional)
- ▶ 4-dimensional momentum space plus one extra dimension: Domain Wall Fermions (algebraic or numerical treatment of 5d)
- ▶ 3-dimensional momentum space plus temporal dimension in position space: Schroedinger Functional implementations

Different classes of regulators

- ▶ 4-dimensional momentum space: Wilson, RHQ, Gauge, Continuum (d -dimensional)

```
// Relativistic Heavy Quark Action
from( " sum(x)*Qb(x)*(
    " + sum(i1,4)*zeta(i1)*Ngamma(i1)*aD(i1,x) + am0 "
    " - sum(i1,4)*r(i1)*aD(i1,i1,x)/2 "
    " + sum(i1,4)*sum(i2,4)*i_*cp(i1,i2)/4"
    " *Nsigma(i1,i2)*aF(i1,i2,x) "
    " )*Q(x) " );
```

- ▶ 4-dimensional momentum space plus temporal dimension:
Dimension of 5d)

- ▶ 3-dimensional momentum space plus temporal dimension in position space: Schroedinger Functional implementations

Different classes of regulators

- ▶ 4-dim

Cont

```
// Domain Wall Action
from( " sum(x)*sum(s,1,N5)*sum(t,1,N5)*Qb(s,x)*(
    "+ ( sum(i1,4)*Ngamma(i1)*aD(i1,x) "
        " - 1/2*sum(i1,4)*aD(i1,i1,x) - aM5Q )*delta(s,t)"
    "+ ( delta(s,t) + d5Q*(
        " - PL*delta(s+1,t) - PR*delta(s-1,t)) )"
    "+ ( PL*delta(s,N5)*delta(t,1) "
        " + PR*delta(s,1)*delta(N5,t))*am0Q"
    ")*Q(t,x) " );
```

- ▶ 4-dimensional momentum space plus one extra dimension:
Domain Wall Fermions (algebraic or numerical treatment of
5d)
- ▶ 3-dimensional momentum space plus temporal dimension in
position space: Schroedinger Functional implementations

Different classes of regulators

- ▶ 4-dimensional momentum space: Wilson, RHQ, Gauge, Continuum (d -dimensional)

► 4-dim
Dom
5d)

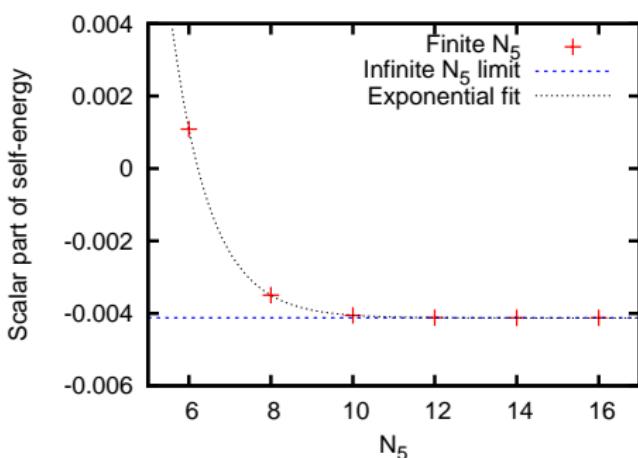
```
// Schroedinger Functional (Wilson)
from( " sum(x,space3)*sum(x0,time)*Qb(pos(x0,x))*( "
    // Wilson
    " + sum(i1,4)*Ngamma(i1)*aD(i1,pos(x0,x)) + am0 "
    " - sum(i1,4)*aD(i1,i1,pos(x0,x))/2 "
    // \delta D_v
    " + sum(i1,4)*sum(i2,4)*i_*csw/4*Nsigma(i1,i2)"
    "                                *aF(i1,i2,pos(x0,x))"
    // \delta D_b
    " + (ct - 1)*("
    "     delta(x0,1) + delta(x0,T-1)"
    " )"
    " )*Q(pos(x0,x)) " );
```

ion:
ent of

- ▶ 3-dimensional momentum space plus temporal dimension in position space: Schroedinger Functional implementations

Example: Domain Wall Fermions

- ▶ Chiral symmetry on the lattice in the limit of large 5d
- ▶ Without chiral symmetry, lattice fermion receives additive mass renormalization



$$m_0 = 0.01, M_5 = 1.3$$

ActionDWF
implements $N_5 \rightarrow \infty$
algebraically

ActionDWFN5
implements finite N_5
numerically

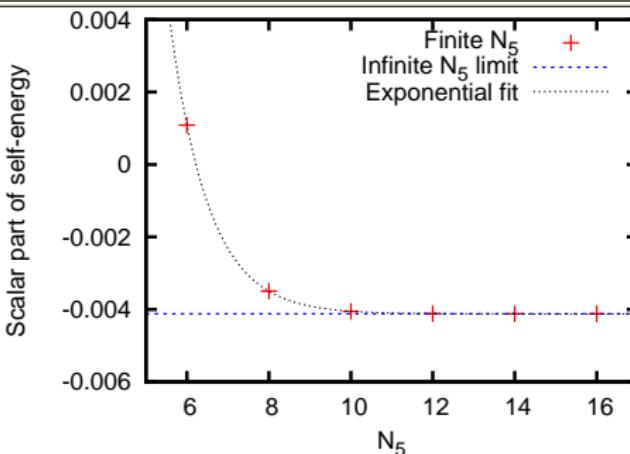
```

// context
Context c;

// lattice setup
ActionDWF dwf(&c, "L", 2);
ActionGAUGE gauge(&c, 1);
// define field rotations (DWF 4d fields)
FieldRotationDWF l(&c, "L", "lmomT", "u^(1/2)*sum(l,1,N5)*"
  "(PL*delta(l,1) + PR*delta(l,N5))*L(l,x)", 0);
FieldRotationDWF lb(&c, "Lb", "lbmomT", "u^(1/2)*sum(l,1,N5)*"
  "Lb(l,x)*(PR*delta(l,1) + PL*delta(l,N5))", 0);
Wick w_lat(&c);
w_lat << dwf << gauge << l << lb;

// create propagator
Expression* ipropDWF = Series::inverted_even(&c,
  w_lat.contract("sum(q,mom)*lmomT(p)*lbmomT(q)",
  2), "gs", 2); // propagator

```



mit of large 5d
on receives additive

$$m_0 = 0.01, M_5 = 1.3$$

ActionDWF
implements $N_5 \rightarrow \infty$
algebraically

ActionDWFN5
implements finite N_5
numerically

Example: perturbative tuning of RHQ action

- ▶ Columbia formulation:

$$S = \sum_x \bar{Q}(x) \left((\gamma_0 D_0 - \frac{1}{2} D_0^2) + \zeta \sum_{i=1}^3 (\gamma_i D_i - \frac{1}{2} D_i^2) + m_0 + c_P \sum_{\mu, \nu=0}^3 \frac{i}{4} \sigma_{\mu\nu} F_{\mu\nu}(x) \right) Q(x)$$

- ▶ Tune coefficients of dimension 4 and 5 operators to remove $|a\vec{p}|$, $(am)^n$, $|a\vec{p}|(am)^n$ errors in on-shell quantities:

$$m_0, \zeta, c_P$$

- ▶ Matching of on-shell matrix elements of, e.g., HL operators requires $Q'(x) = Q(x) + d_1 \sum_{i=1,2,3} \gamma_i D_i Q(x)$ with parameter d_1 .
- ▶ Tune ζ by matching propagator, tune c_P by matching quark-gluon vertex

Excerpt of RHQ tuning code

```
Context c;

// use rhq + gauge action
ActionRHQ rhq(&c, "Q");
ActionGAUGE gauge(&c);

// define field rotations
c.coefficients << "d1FT";
FieldRotationRHQ Qimp(&c, "Q", "QimpmomT",
    "(1 + sum(i,4)*d1FT(i)*Ngamma(i)*aD(i,x))*Q(x)");
FieldRotationRHQ Qbimp(&c, "Qb", "QbimpmomT",
    "Qb(x)*(1 - sum(j,4)*d1FT(j)*Ngamma(j)*aDl(j,x))");

// perform wick contractions
Wick w(&c);
w << rhq << gauge << Qimp << Qbimp;
Expression* vertex = w.contract(
    "sum(k,mom)*QimpmomT(q)*aACmom(mu1,a1,k)*QbimpmomT(-p)", 3);
Expression* prop = w.contract(
    "sum(q,mom)*QimpmomT(p)*QbimpmomT(q)", 2);
```

Excerpt of continuum code

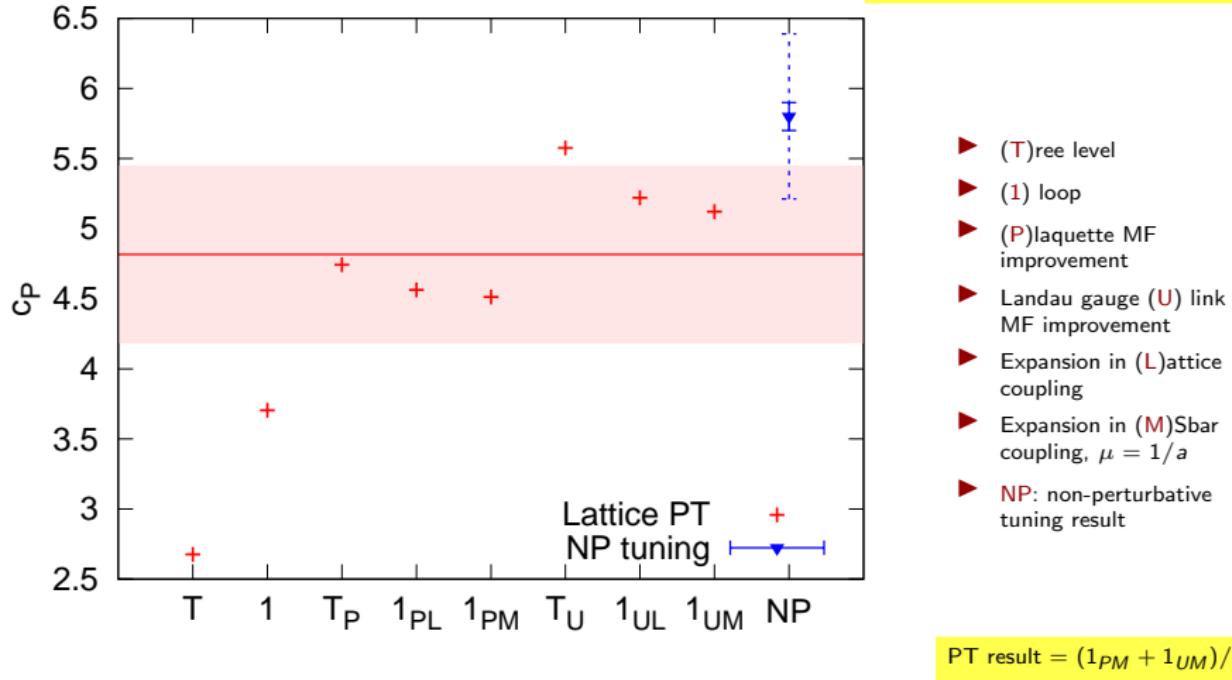
```
// context
Context c;

// continuum setup
ActionCONTQ contq(&c, "QC");
ActionCONTG contg(&c);
Wick w_con(&c);
w_con << contq << contg;

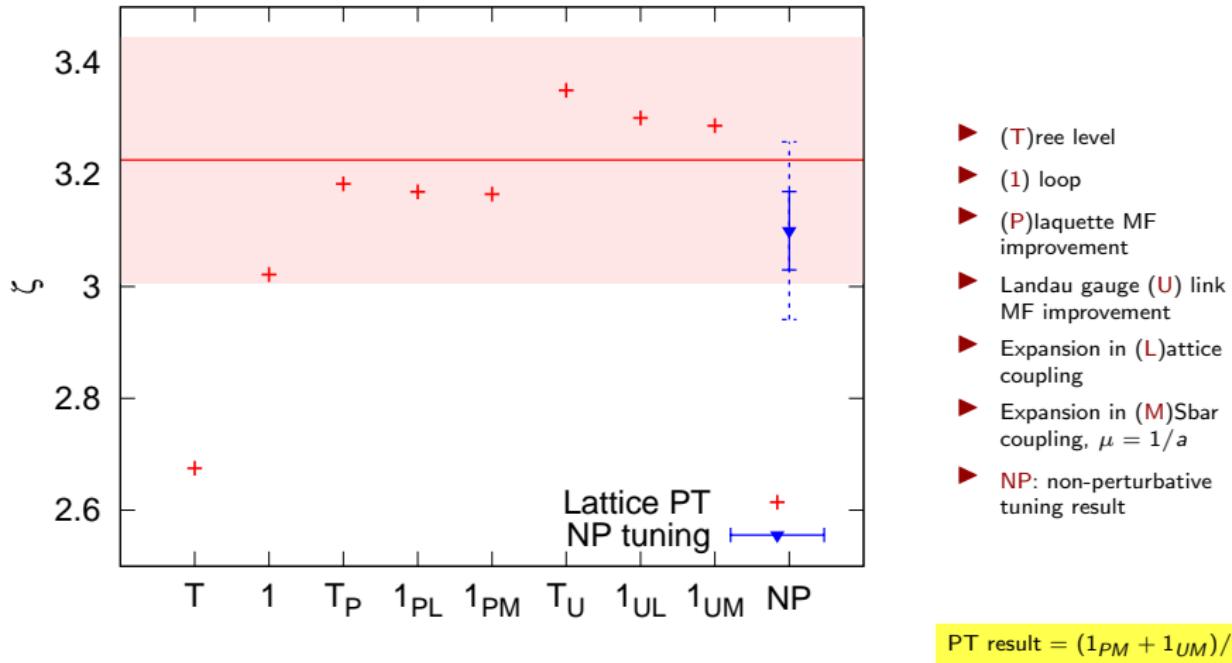
// create vertex
Expression* vertexCON =
    w_con.contract_abstract("sum(k,mom)*QCmomT(q)"
    "*aACmom(mu1,a1,k)*QCbmomT(-p)",3);

// amputate vertex
...

// integrate using dim reg
DimReg(&c, vertexCON).prepare().combineFermionLines()
    .onshell("p", "ampQC", "q", "ampQC").combineFermionLines()
    .separateIntegrals().tensorToScalarIntegrals()
    .reduceToMasterIntegrals()
    .insertMomentumConfiguration(onShellConf)
    .expandInEpsilon(0).flatten();
```



- PT error is maximum of naive $\alpha_s^2 \sim 5\%$ error and $(1_{UM} - 1_{PM})$
- NP error is stat. (inner) and stat. + syst. in quadrature (outer)



- PT error is maximum of naive $\alpha_s^2 \sim 5\%$ error and $(1_{UM} - 1_{PM})$
- NP error is stat. (inner) and stat. + syst. in quadrature (outer)

Conclusion and Outlook

- ▶ New framework for lattice and continuum PT (CL PoS LATTICE 2012, more details published soon)
- ▶ First applications:
 - ▶ Tuning of RHQ action
 - ▶ Matching and $O(a)$ improvement of heavy-heavy, light-light, and heavy-light bilinears (light DWF)
 - ▶ Tree-level matching and $O(a)$ improvement of mixing matrix elements
- ▶ Outlook:
 - ▶ Two-loop matching
 - ▶ Tuning of highly-improved actions
 - ▶ Implementation of further regulators (and, e.g., Gradient Flow)